2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Structured Motion Generation with Predictive Learning: Proposing Subgoal for Long-Horizon Manipulation

Namiko Saito[1,2,3], João Moura [1,2], Tetsuya Ogata[4], Marina Y. Aoyama[1], Shingo Murata[5], Shigeki Sugano[3] and Sethu Vijayakumar[1,2]

***Abstract*— For assisting humans in their daily lives, robots need to perform long-horizon tasks, such as tidying up a room or preparing a meal. One effective strategy for handling a long-horizon task is to break it down into short-horizon subgoals, that the robot can execute sequentially. In this paper, we propose extending a predictive learning model using deep neural networks (DNN) with a Subgoal Proposal Module (SPM), with the goal of making such tasks realizable. We evaluate our proposed model in a case-study of a long-horizon task, consisting of cutting and arranging a pizza. This task requires the robot to consider: (1) the order of the subtasks, (2) multiple subtask selection, (3) coordination of dual-arm, and (4) variations within a subtask. The results confirm that the model is able to generalize motion generation to unseen tools and objects arrangement combinations. Furthermore, it significantly reduces the prediction error of the generated motions compared to without the proposed SPM. Finally, we validate the generated motions on the dual-arm robot Nextage Open. See our accompanying video here: https://youtu.be/3hYS2knRm5o**

I. Introduction

Robots capable of performing daily activities, such as cooking, will have to reason about long-horizon manipulation tasks [1], [2]. One effective strategy of handling a long-horizon task (goal) is to break it down into subtasks (subgoals) and then organize them in a way that leads to the desired outcome. For instance, cooking combines the subtasks of peeling, cutting, stirring and so on, to achieve the final goal of preparing a meal. Hence, the key question we address in this work is: given a final goal requiring long-horizon manipulation, how can robots propose subgoals, without human assistance, to improve task completion?

There are several challenges with automatically setting subgoals to achieve long-horizon manipulation tasks, that require multiple subtasks namely conduct grasping and releasing objects more than three times. First, the robot needs to consider the order of the subgoals to pursue. For example, we typically wash, then peel, then chop the vegetables. Second, for each stage of the task, the robot has to select the appropriate subgoal from multiple possible subgoals. For instance, we select the vegetable knife instead of the bread one to chop the vegetables, even though both of them are possible choices. There are other works addressing this specific topic of tool selection [3]–[6]. Third, at each stage, the robot also needs to select the appropriate arm or both for the subtask, which will depend on reachability and avoiding interference between the arms [7]. Finally, due to the interactive nature of manipulating objects, there is some stochasticity associated with each individual subtask. Therefore, for each subtask, the robot also needs to adjust the motion to slight variations in the locations of the objects/tools as well as the timing of each action.

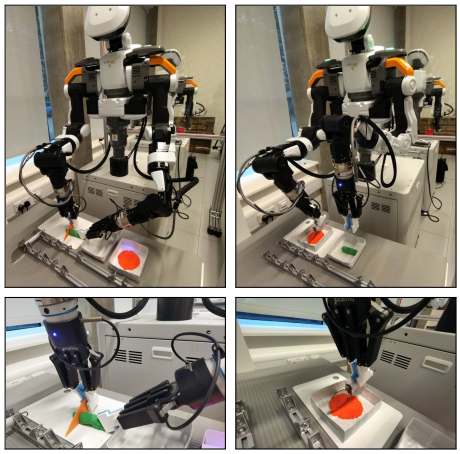

Fig. 1: Dual-arm robot cutting a pickle with a knife, and a pizza with a pizza cutter.

A. Related Work

Lynch et al. [8] and Nair et al. [9] tackle the challenge of achieving a manipulation task composed of several subtasks. Lynch et al. [8] uses the pairs of current and goal images to estimate the entire sequence of actions to achieve the goal, by encoding the task in a latent space. Nair et al. [9] also uses difference between a current image and a goal image to infer the latent space, but to select the following action. Florensa et al. [10] used Reinforcement Learning to learn all feasible goal paths in the environment for locomotion. However, all the works encode different subtasks as a single

*This research has received funding from European Union's Horizon 2020 research and innovation programme under grant agreement No 101017008, Enhancing Healthcare with Assistive Robotic Mobile Manipulation (HARMONY), JSPS Grant-in-Aid for Scientific Research (A) No. 19H01130, JST Moonshot R & D Grant Number JPMJMS2031, Waseda Research Institute for Science and Engineering, Grant-in-Aid for Young Scientists (Early Bird), and Kayamori Foundation of Informational Science Advancement. This research is supported by Kawada Robotics Corporation, Japan and the Alan Turing Institute, UK.

[1]Authors are with the School of Informatics, The University of Edinburgh, Edinburgh, U.K.

[2]Authors are with The Alan Turing Institute, London, U.K.

[3]Authors are with Department of Modern Mechanical Engineering, Waseda University, Tokyo, Japan. n_saito@sugano.mech.waseda.ac.jp

[4] Author is with the Department of Intermedia Art and Science, Waseda University, Tokyo, Japan, and the National Institute of Advanced Science and Technology, Tokyo, Japan.

[5]Author is with Department of Electronics and Electrical Engineering, Keio University, Kanagawa, Japan.

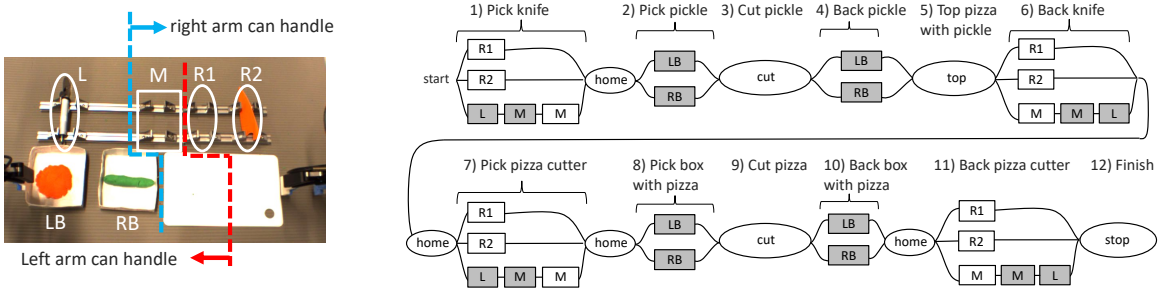

Fig. 2: The table setting through the robot eye camera, and the whole flow of the long-horizon cutting and arranging tasks. In the right figure, the tasks of gray boxes require to be conducted by left arm, the white boxes by right arm, and the white circles by dual arms.

task and, therefore, only tackle short sequences of subtasks.

Lin et al. [11] propose a method for learning a sequence of subtasks from demonstration. They use a Variational Auto Encoder (VAE) to infer a latent vector that represents the subtasks in the latent space as intermediate goals. Pertsch et al. [12] also learn a sequence of subtasks where, additionally, they handle the possibility of multiple following subtasks by using a distance metric in the latent space. Other similar approaches of handling the multiple choice of subtask are [13], [14]. However, these methods focus on the high level reasoning of the subgoals, which ignores aspects of the variations within the subtasks.

Junge et al. [15] addressed the problem of slight variations in the time duration of each of the subtasks by optimizing learnt parameters that relate with the subtask duration. Wang et al. [16] and Holladay et al. [17] both address the challenge of uncertain position of the object being manipulated. They propose methods for generating robot motions based on the confidence of the estimated object pose. Additionally, Holladay et al. [17] took into account force constraints. All the previous works focus on adaptations within a subtask, given a pre-specified sequence of subtasks.

B. Problem Description

Previous works fail to address very long-horizon manipulation tasks, requiring many intermediate subtasks, and the use of dual-arm robots, which increases the combinations of possible motions. Therefore, we aim to address the challenge of generating long-horizon manipulation motions for dual-arm robotic tasks, for which the introduction of subgoals can benefit the task completion. For that, we need to consider: (1) the order of the subtasks, (2) subtask selection, (3) coordination of dual-arm, and (4) variations within a subtask.

As case study, we use the task of cutting and arranging a pizza, whose assembly order is strict and needs multiple grasping subtasks. Fig. 1 shows the dual-arm robotic setup, and Fig. 2 shows the workspace board and the task flow. The task consists in a robot picking a pickle and fixing on the cutting board, cutting a pickle with a vegetable knife, followed by placing a piece of it on a pizza dough and returning the remaining to the storage box. Then picking the box with pizza and moving it to the cutting board, cutting the pizza with a pizza cutter, and finally placing the box and knives in their initial positions. We increase the number of combinations of possible motions by: having different possible initial placements for the knives and the ingredients, including being on the left side of the workspace, requiring the robot to perform handover of the knives to be able to always cut with the right arm. Given the interactive nature of the task, the ingredients and knives inevitably end up in slightly different positions throughout the task. Therefore, the robot needs to consider: proper order to act, proper tool selection, dual arm coordination, and adaptation to variations in objects' positions.

C. Contribution

In previous work [6], we proposed a DNN model consisting of a Convolutional Autoencoder (CAE) [18] and a Multiple Timescales Recurrent Neural Network (MTRNN) [19] for learning subtask selection and adaptation within subtask variations. We applied our method to the problem of tool selection in a stirring and pouring scenario according to the ingredient properties. The CAE transforms images into small dimensional features, while the MTRNN predicts next time sensorimotor data and generates motions. We showed that the MTRNN can integrate multimodal information, and the robot could recognize tool-object relations from the visual and tactile information. Additionally, the MTRNN's slow and fast context nodes allow it to learn fairly long time-series data. However, we will show that for demonstrations with longer time horizons and number of subtasks, such as the ones described in the Section I-B, the prediction error increases significantly.

In this work we improve the previous DNN model by adding a Subgoal Proposal Module (SPM), which can propose subgoals to the MTRNN. At each time step, the SPM predicts the following subgoal only by using the current and the final goal images. We train both the SPM and MTRNN by demonstration providing examples of the subgoals, which follow a specific rule to break the long-horizon task. During the test, the MTRNN generates robot motions using the subgoals predicted by the SPM. We evaluate our proposed method using the task of cutting and arranging pizza described in the Problem Description using the Nextage open [20], where we test if the model can generalize to unseen arrangement combinations of tools and food objects. We compare the proposed method against our previous work, to confirm the effectiveness of the SPM in improving motion

generation. In summary, our contributions are as follows:

- Proposing a method for learning long-horizon robot motions with several possible subtasks, by adding a Subgoal Proposal Module (SPM) to the CAE and MTRNN architecture.
- Demonstrating that the encoding of the subgoals allows the learnt model to generalize to unseen arrangement combinations of tools and objects being manipulated.
- Validating the generated motions in a physical dual-arm robotic setup.

II. Method

There are three steps for constructing the DNN model: demonstration data collection, indicating subgoals of training data, and training DNN modules.

A. Demonstration Data collection

For collecting the demonstration data, we control the robot by teleoperation or replay of keyposes, for the whole cutting and arranging process, and record time-synchronized sensorimotor data, consisting of image, force, torque, end-effector position, and gripper width. For teleoperation, we use a 3D mouse controller to command the robot during cutting, and pick and place subtasks. We use keyposes for sending the robot to specific pre-specified positions, such as above the knives and food objects.

B. Indicating subgoals

We set subgoals for training data according to the following the rules: the robot (a) changes the gripper width, (b) arrives to the home position and (c) starts or finishes cutting, which can divide the long-horizon task to short subtasks. These rules correspond to the switches between different types of actions; for example from a tool-selection subtask to grasping subtask, from a tool bringing subtask to a towarding to the board subtask, and from a towarding to the board subtask to a cutting subtask. We hypothesise that, after training, the SPM will be able to propose subgoals according to the rules.

In our task, we indicate subgoals with images because they can show the situations of workspace and tell the actions to be conducted, that is one of the simplest ways to label the subgoals.

C. Deep learning modules

Fig. 3 shows the overall DNN model, with the CAE and the MTRNN, from our previous work [6], and the additional SPM. There are 2 stages to train the DNN model. The first is training the CAE. Then simultaneously training the SPM and the MTRNN, while fixing the weights of the CAE.

1) CAE: We trained the CAE [18] to make the output image data the same as the input image data. Then 30 nodes of the intermediate layer extracts the image features ($y^{\mathrm{ImgFeature}}(t)$). Table I shows the construction. As the activation function, we use sigmoid for the middle layer and ReLU for the others, with kernel 4×4, stride 2 and zero padding 1. The module is trained 1000 epochs and thereafter, we add noise of brightness, parallel translation and rotation to the training image data and train additional 100 epochs.

TABLE I: Structure of the CAE and SPM (till line 7)

	Input	Output	Processing
1	(96, 64, 3)	(48, 32, 32)	convolution
2	(48, 32, 32)	(24, 16, 64)	convolution
3	(24, 16, 64)	(12, 8, 128)	convolution
4	(12, 8, 128)	(6, 4, 256)	convolution
5	6144	600	fully connected
6	600	200	fully connected
7	200	30	fully connected
8	30	200	fully connected
9	200	600	fully connected
10	600	6144	fully connected
9	(6, 4, 256)	(12, 8, 128)	deconvolution
10	(12, 8, 128)	(24, 16, 64)	deconvolution
11	(24, 16, 64)	(48, 32, 32)	deconvolution
12	(48, 32, 32)	(96, 64, 3)	deconvolution

2) SPM: We construct the SPM encoder, for outputting subgoal features using final goal and current image, whose idea refers [9]. For training the module, we use indicated ground truth subgoals mentioned in Section II-B. At first, the images of indicated subgoals are converted to image features by the CAE, which is ground truth subgoal image features ($y^{\mathrm{TruthSubFeature}}(t)$). Then, we train the SPM inputting difference between a current image ($x^{\mathrm{Img}}(t)$) and the final goal image which is given in advance, and outputting subgoal image features ($y^{\mathrm{SubFeature}}(t)$). The module is trained minimizing the error as

$$E=\frac{1}{N}\sum\left(y^{\mathrm{SubFeature}}(t)-y^{\mathrm{TruthSubFeature}}(t)\right)^2. \quad (1)$$

The construction and setting are the same as the encoder part of the CAE, shown in Table I.

3) MTRNN: We use the MTRNN to integrate all data and predicts the next step from the current and context data. Because of their large time constant (set 30, whose length is almost the same as a subtask), slow context ($\mathrm{C_s}$) nodes learn data sequences, whereas fast context ($\mathrm{C_f}$) nodes with a small time constant (set 5) learn motion primitives. We set the number of $\mathrm{C_s}$ nodes 50 and $\mathrm{C_f}$ nodes 150.

The input of the MTRNN ($x(t)$) is image features by the CAE, subgoal features by the SPM, end-effector position and gripper width, and force / torque sensor data. The forward calculations of the internal value u_i is as

$$u_i(t)=\left(1-\frac{1}{\tau_i}\right)u_i(t-1)+\frac{1}{\tau_i}\left[\sum_{j\in\mathrm{n}}w_{ij}x_j(t)\right], \quad (2)$$

where n is the number of neurons, τ_i is the time constant, and w_{ij} is the weight. The output is then calculated as

$$y_i(t)=\tanh\left(u_i(t)\right). \quad (3)$$

The robot is controlled using the output end-effector position and gripper width. The value of $y_i(t)$ is then used as the next input value as

$$x_i(t+1)=\begin{cases}\beta\times y_i(t)+(1-\beta)\times \mathrm{T}(t+1) & i\in \mathrm{IO}\\ y_i(t) & \text{otherwise}\end{cases}, \quad (4)$$

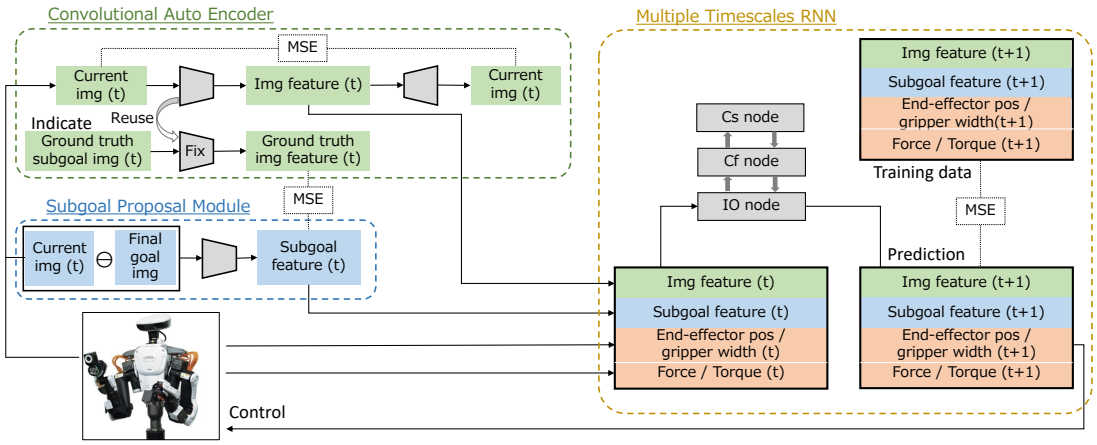

Fig. 3: Overall learning model. The CAE compresses image data to low-dimensional image features. The SPM proposes subgoals by inputting the current and the final goal image. The MTRNN learns to generate the motions following subgoals.

where IO is Input-Output layer, and $\mathrm{T}(t)$ is the input datum, which is training data-set during training, whereas test data-set or real time data during evaluation experiments. The next input value $x_i(t+1)$ is adjusted by multiplying the output of the preceding step $y_{\mathrm{IO}}(t)$ and the datum $\mathrm{T}(t+1)$ by the feedback rate $\beta(=0.5)$. If the feedback rate is large, the model stably predict the next step, whereas if it is small, the model easily adapt to the input.

For the backward calculations, we add the error by SPM (Eq. (1)) and MTRNN (Eq. (5)) and use back propagation through time (BPTT) algorithm [21] to minimize it to update SPM and MTRNN at the same time,

$$E = \alpha \sum_{t} (y_{\mathrm{IO}}(t-1) - \mathrm{T}(t))^2 . \qquad (5)$$

We train the SPM and the MTRNN for 2000 epochs in a well-balanced manner by adjusting the parameter $\alpha(=1/50,000)$ to make the amount of the error of MTRNN similar to the one of SPM.

III. Experimental Setup

A. Hardware design and data sampling

In the robot experiment, we use Nextage open, which has 6 DOF arms and 2 cameras on its head, and we use the left one. We attach force and torque sensors named F/T Sensor gamma and Robotiq 2f-140 grippers to its both wrists.

We record robot end-effector xyz-position and quaternion of both arms (7 dim x 2), gripper width (1 dim x 2), force and torque data (6 dim x 2), and RGB images (96 (width) $\times$ 64 (height) $\times$ 3 (channels)) every 0.2 sec. The lengths of motions are 118.6 - 155.0 sec, thus they are 593 - 775 steps. The values of image data is normalized to [0, 255] and other sensorimotor data is normalized to [-0.9, 0.9] before inputting to the model.

B. Object used and data variance

For experiments, we prepared an orange plastic vegetable knife and a white pizza cutter. As for food objects, we made the pickle whose length is 11 cm with 85 g green clay, and the pizza whose diameter is 9 cm with 70 g orange clay.

TABLE II: Tool and object arrangement, with P: pizza, p: pizza cutter, K: pickle, k: vegetable knife.

Object												
LB	P	P	K	K	P	P	K	K	P	P	K	K
RB	K	K	P	P	K	K	P	P	K	K	P	P
Tool												
L	k	-	k	-	k	-	k	-	p	p	p	p
R1	-	k	-	k	p	p	p	p	k	-	k	-
R2	p	p	p	p	-	k	-	k	-	k	-	k
Leave untrained for evaluation												
	1	-	-	-	-	-	-	1	1	-	-	1

The table setting is shown in Fig. 2. We put a tool holder, 2 squared boxes (LB, RB) and a cutting board on a table. There are 4 places to put tools on the tool holder; L, M, R1, R2. Both arms can reach the place M, which is used for passing a tool from left to right hand or vice versa. Thus there are 3 initial positions for 2 tools, in other words 6 combinations of tool arrangement. Whereas each food object is put on the boxes, thus there are 2 combinations of initial object arrangement. Thus there are 12 combinations (6 x 2) of arrangements.

We used 8 different arrangements of tools and food objects for training and left 4 for evaluation, given a total of 12 arrangement combinations, as shown in Table II. The 8 training arrangements cover all subgoals, to make sure the DNN model experiences all possible actions. For each arrangement of tools and objects we collected 3 demonstrations, resulting in a total of 24 training data-sets. We collected all the data-sets in advance, including the 4 unknown arrangement combinations of tools and objects left for evaluation.

C. Evaluation

We validate if our model can predict subgoals according to the rules in Section II-B and generate motions, using the 4 test arrangement combinations. Furthermore, we compare the results of our model with the baseline, without the SPM. Finally, we experimentally test the motion generation with the physical robot, using either data collected offline or in an online control loop, i.e. with sensory data being generated online. Hence, the input datum $\mathrm{T}(t)$ in Eq. (4) originated

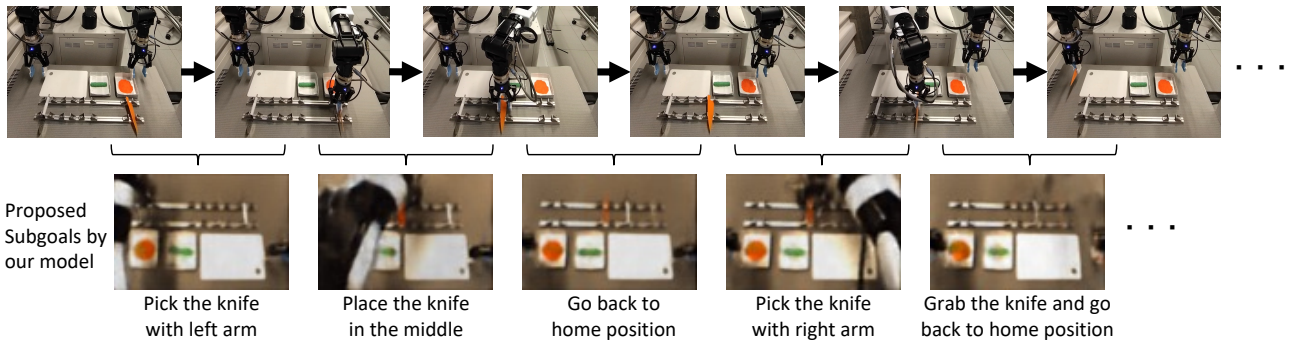

Fig. 4: Playing back of the motion generated by the MTRNN model, with untrained arrangement of the tools and objects, where the robot passes the knife from left to right. The bottom pictures show the decoded images of subgoal features proposed by the SPM as the following subgoals.

either from a pre-recorded offline data-set (offline test) or from online sensor readings (online experiment).

Decision of success or failure is based on the 3 factors; (1) order of actions, (2) tool and object association, and (3) use the correct arm (right, left and, dual arm(s)). We check if the robot can conduct subtasks which matches all the factors.

IV. Results and Discussion

In this section we present the results of evaluating both the SPM and the MTRNN, using offline motion generation, and discuss the case of online motion generation.

A. Analysis of SPM and proposed subgoals

For evaluating that the SPM can propose correct subgoals, we decoded the subgoals' features, generated for an unseen data-set, and reconstructed the images using the decoder of the CAE. The bottom row in Fig. 4 shows proposed subgoal images produced while passing the knife from left to right. The arms' configuration and corresponding timing shown in the proposed subgoal images follows the rules described in Section II-B, indicating that the SPM was able to capture the rules only from training data.

B. Analysis of MTRNN and motion generation

We compared the predicted end-effector positions generated by the MTRNN with (ours) and without SPM (baseline) for the test data. Fig. 5 shows that our model generates end-effector positions with significantly smaller error than the baseline, specially towards the final part of the long-horizon trajectory. Fig. 6 shows the median and 75 percentile errors of the prediction of the end-effector positions and angles for the full test data-set, which confirm the trends in Fig. 5. Furthermore, we streamed the motion generated by our model in the physical robot hardware, accomplishing the desired goal, which would have been impossible with the baseline, given the significant position and orientation errors. The physical robot succeeded in accomplishing all the 4 test arrangements, fulfilling correct (1) order of actions, (2) tool and object association, and (3) use of the correct arm. The upper row of Fig. 4 shows a part of the task.

To analyze internal state of the MTRNN and check if the prediction could follow subgoals, we conducted PCA on Cs nodes, which take a role of memorizing long sequential data

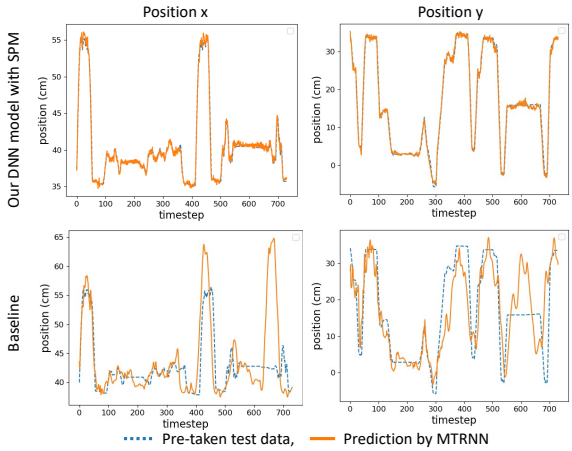

Fig. 5: Each column corresponds to the output x and y position trajectory of the right hand, generated by the MTRNN using test data. The upper plots show our model predicted trajectories and the bottom plots show the baseline, without using the SPM.

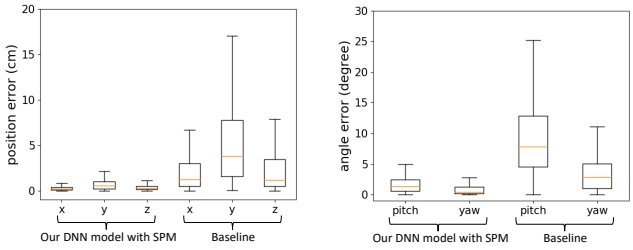

Fig. 6: The median and 75 percentiles of the prediction error for the position and pitch and yaw angles (computed from quaternions) of the right hand, using test data. Comparison between our model and the baseline (without SPM).

and direct overall actions. The left and right columns of Fig. 7 show the PCA result for different arrangements of the tools and objects, where the numbers correspond to the order of subgoals and the colors to different types of action. Fig. 7 shows that, for our model and for both arrangements, the path of the subgoals aligns according to the type of action. For example, with our model, the subgoals related with the cutting of the pizza and the cutting of the pickle cluster in separate areas of the plot. This is an indication

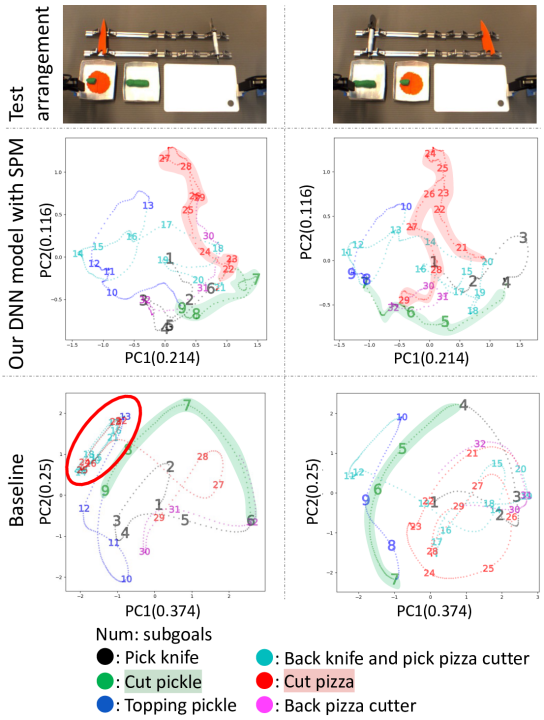

Fig. 7: PCA on sequential internal value of Cs nodes in MTRNN comparing our model and the baseline. The numbers correspond to the order of subgoals and the colors correspond to different types of action. Each column corresponds to different test arrangements of the tools and objects.

that the Cs nodes structurally represent the sequential nature of the subgoals that the robot follows. In the case of the baseline (bottom plots), the Cs nodes appear to be much less sequentially organized. For example, the subgoals related with cutting the pizza spread all over the plot. In Fig. 7, we highlight with a red ellipse a repetition of the flow, which might explain the repeating pattern of the baseline trajectory in Fig. 5. We concluded the SPM critically supports MTRNN for motion generation by sequentially proposing subgoals.

C. Online motion generation

The highly interactive nature of our case study, which includes motions such as picking up tools and cutting objects, makes reproducing motions on a physical robot quite challenging. On one hand, those tasks require quite precise control of the robot's end-effectors; and on the other hand, those interactions can change the objects positions in unpredictable ways. So far, we discussed the results corresponding to the case where we were able to use our learnt model to offline generate a full motion and replay it in the physical robot, which would have been unattainable with the baseline model (see position error in Fig. 5). In this section we investigate the case of deploying the learnt model in an online fashion, i.e. each control step takes the sensory data (camera, force/torque, position) from the physical robot and predicts the next robot position using the learnt model.

When deploying our learnt predictive model in an online control loop, we realized that there is a small and incremental growth in the deviation of the robot motion. That compound error propagation is significant enough that by the time the robot attempts to pick up the tool for the second time (corresponding to step 7 in Fig. 2), we have displacements on the order of 1.5 cm in the xy-plane, consistently making the robot fail the grasping. The accumulation of the error in the offline case is less critical because the update of the next state uses sensory information that is similar to the one used for learning the model itself. Moreover, the online experiments took place in a different season from the data collection, which resulted in significant changes in the lighting conditions. Furthermore, since we collected trajectories (using keyposes with added noise) with a fairly low variation across demonstrations, the training data fails to cover a large portion of the workspace. Therefore, during the online experiments, once the robot starts experiencing different sensory information from the training data, the error in the prediction increases further, both in the end-effector's pose and time of reach. The large range of motion in our task and the need to synchronise commands of several modalities/dimensions (such as position, orientation, and time of grasping) also aggravate the problems previously described. Although we are able to construct a predictive model for offline use, we conclude that the problem of online precise control of robots for long-horizon tasks raises additional challenges, related with the incorporation of DNN models in closed-loop systems.

V. Conclusion

In this paper, we proposed a DNN model which realizes long-horizon motion task generation considering (1) the order of the subtasks, (2) subtask selection, (3) coordination of dual-arm, and (4) variations within a subtask. The model contains: the CAE which compresses images to low dimensional features; the Subgoal Proposal Module (SPM) which predicts following subgoal, from difference between current and final images; and the MTRNN which generates a motion according to the proposed subgoal. We tested motion generation experiments with untrained tool and object arrangement combinations, and confirmed the effectiveness of subgoal proposal by comparing it against the baseline without SPM.

As we discussed in Section IV-C, there are limitations in precise manipulation and robustness to online motion generation. To address this issue, we will investigate the combination of DNN with motion planning, such that the DNN generates higher-level decision and the motion planner generates lower-level precise control.

In this task, we indicated the subgoal with images, which can tell the situations and actions needed, however other ways to label subgoals would be useful for other tasks. The next work would be implementing other logical way to label subgoals for other variety of tasks. In addition, our current method requires a supervised learning phase where we provide examples of the ground truth subgoal images. For future work, We would also like to investigate the application of unsupervised learning methods for automatically extracting subgoal images from sequential videos.

REFERENCES

[1] K. Yamazaki, R. Ueda, S. Nozawa, M. Kojima, K. Okada, K. Matsumoto, M. Ishikawa, I. Shimoyama and M. Inaba, "Home-Assistant Robot for an Aging Society," Proceedings of the IEEE, vol. 100, no. 8, pp. 2429–2441, 2012.

[2] G. Kazhoyan, S. Stelter, F. K. Kenfack, S. Koralewski and M. Beetz, "The Robot Household Marathon Experiment," Proceeding of 2021 IEEE International Conference on Robotics and Automation, pp.9382–9388, 2021.

[3] A. Xie, F. Ebert, S. Levine and C. Finn, "Improvisation through Physical Understanding: Using Novel Objects as Tools with Visual Foresight," Robotics: Science and Systems, 2019.

[4] K. P. Tee, J. Li, L. T. P. Chen, K. W. Wan, G. Ganesh, "Towards Emergence of Tool Use in Robots: Automatic Tool Recognition and Use without Prior Tool Learning," Proceeding of 2018 IEEE International Conference on Robotics and Automation, pp.6439–6446, 2018.

[5] K. Fang, Y. Zhu, A. Garg, A. Kurenkov, V. Mehta, L. Fei-Fei, and S. Savarese, "Learning task-oriented grasping for tool manipulation from simulated self-supervision," The International Journal of Robotics Research, pp. 202–216, 2020.

[6] N. Saito, T. Ogata, S. Funabashi, H. Mori and S. Sugano, "How to Select and Use Tools? : Active Perception of Target Objects Using Multimodal Deep Learning," IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 2517–2524, 2021.

[7] S. Murata, Y. Li, H. Arie, T. Ogata and S. Sugano, "Learning to Achieve Different Levels of Adaptability for Human–Robot Collaboration Utilizing a Neuro-Dynamical System," IEEE Transactions on Cognitive and Developmental Systems, vol. 10, no. 3, pp. 712–725, 2018.

[8] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine and P. Sermanet, "Learning Latent Plans from Play," Proceeding of Conference on Robot Learning, 2019.

[9] S. Nair, S. Savarese and C. Finn, "Goal-Aware Prediction: Learning to Model What Matters," Proceeding of the 37th International Conference on Machine Learning, pp. 7207–7219, 2020.

[10] C. Florensa, D. Held, X. Geng, and P. Abbeel. "Automatic goal generation for reinforcement learning agents." International conference on machine learning, pp. 1515–1528, 2018.

[11] X. Lin, Z. Huang, Y. Li, J. Tenenbaum, D. Held and C. Gan, "DiffSkill: Skill Abstraction from Differentiable Physics for Deformable Object Manipulations with Tools," Proceeding of International Conference on Learning Representations, 2022.

[12] K. Pertsch, O. Rybkin, F. Ebert, C. Finn, D. Jayaraman, S. Levine, "Long-horizon visual planning with goal-conditioned hierarchical predictors," Proceeding of the 34th International Conference on Neural Information Processing Systems, pp.17321–17333, 2020.

[13] B. Ichter and M. Pavone, "Robot Motion Planning in Learned Latent Spaces," IEEE Robotics and Automation Letters, vol. 4, no. 3, pp.2407–2414, 2019.

[14] B. Brito, M. Everett, J. P. How, and J. Alonso-Mora, "Where to go Next: Learning a Subgoal Recommendation Policy for Navigation in Dynamic Environments," IEEE Robotics and Automation Letters, vol. 6, no. 3, pp.4616–4623, 2021.

[15] J. Yi, T. A. Luong, H. Chae, M. S. Ahn, D. Noh, H. N. Tran, M. Doh, E. Auh, N. Pico, F. Yumbla, D. Hong and H. Moon, "An Online Task-Planning Framework Using Mixed Integer Programming for Multiple Cooking Tasks Using a Dual-Arm Robot," Applied Sciences vol. 12, no. 8, 2022.

[16] Z. Wang, C. R. Garrett, L. P. Kaelbling and T. Lozano-Pérez, "Learning compositional models of robot skills for task and motion planning," The International Journal of Robotics Research, vol. 40 (6-7) pp. 866–894, 2021.

[17] R. Holladay, T. Lozano-Pérez and A. Rodriguez, "Robust Planning for Multi-stage Forceful Manipulation," arXiv:2208.00319, 2022.

[18] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," Science, vol. 313, no. July, pp. 504-507, 2006.

[19] Y. Yamashita and J. Tani, "Emergence of Functional Hierarchy in a Multiple Timescales Recurrent Neural Network Model: A Humanoid Robot Experiment," PLoS Computational Biology, vol. 4, no. 11, 2008.

[20] Kawada Robotics, "Next generation industrial robot Nextage," http://www.kawadarobot.co.jp/en/nextage/

[21] P. J. Werbos, "Backpropagation through Time: What It Does and How To Do It," Proceeding of the IEEE, vol.78, no.10, pp. 1550–1560, 1990.